STOCKHOLMS
UNIVERSITET

Remote Camera Controller Implemented on the Canon VC-C4

Ahmed Abdela

TRITA-NA-E05093

Numerisk analys och datalogi
KTH
100 44 Stockholm

Department of Numerical Analysis
and Computer Science
Royal Institute of Technology
SE-100 44 Stockholm, Sweden

Remote Camera Controller Implemented on the Canon VC-C4

Ahmed Abdela

TRITA-NA-E05093

Bachelor's Thesis in Computer Science (10 credits)
Single Subject Courses,
Stockholm University 2005
Supervisor at Nada was Alex Jonsson
Examiner was Stefan Arnborg

Abstract

This thesis presents the design and implementation of a remote camera controlling application for Canon VC-C4 and VC-C4R, The application is implemented in low level programming language for MS Windows 95 and above.

The project consists of two programs, a Client and a Server.

Remote VCC4 uses a simple interface on both Client and Server. The end-user can control the camera from a remote computer, change focus, zoom, flicker, speed et cetera.

Sammanfattning

Fjärrstyrning av kamera, implementerad på Canon VC-C4.

Examensarbetet presenterar design och implementering av en fjärrstyrd kameraapplikation, för Canon VC-C4 och VC-C4R (Remote VC-C4). Applikationen är implementerad i ett lågnivåprogrammeringsspråk för MS Windows 95 och senare.

Projektet består av två program, en klient och en server.

Remote VCC4 använder ett enkelt gränssnitt på både klient and server. Slutanvändaren kan kontrollera kameran från en fjärrdator, ändra fokus, zoom, flimmer, hastighet osv.

DEDICATED TO MY BELOVED FAMILY

Acknowledgments

I would like to thank my teachers for helping me achieve what I have achieved, also I like to thank the Staff at KTH, NADA(The Royal Institute of Technology, Department of Numerical Analysis and Computer Science) for helping me out with my project and for being so kind and patient.

Special thanks goes to Mr. Mats Erixson at KTH, AMT(Advanced Media Technology) and Dr. Alex Jonsson at KTH, GT (Graphic Arts Technology and Management).

Organization of the Master's thesis

Chapter 1
Is a basic description of the problem at hand, followed by an outline of the solution to be implemented.

Chapter 2
Is about the planning phase and a review of tools, on which this project will be implemented.

Chapter 3
Is the heart of the thesis, where the implementation method is carried out, and where the user interface is decided. In a nutshell this chapter is about the implementation in detail.

Chapter 4
Is the testing phase, where test equipment is presented and test results are given.

Chapter 5
Is a conclusion of the whole project.

Contents

1 Introduction **1**
1.1 Description of the problem 1
1.2 Overview of the client 1
1.3 Overview of the server 3

2 Planning phase **4**
2.1 Tools 6
2.1.1 Compiler MASM32 6
2.1.2 IDE RADasm 7

3 Implementation phase **8**
3.1 TCP/IP 8
3.2 Serial port 10
3.3 Server 11
3.4 Client 12

4 Test phase **14**

5 Conclusion **15**

References **16**

Appendix: Control Command Tables **17**
Pedestal Control Command Table 17
Camera Control Command Table 18
System control Command Table 19

1 Introduction

The idea is to control a camera connected to a computer serial port, from a remote client computer, connected via any IP based network.

The hardware equipment are a Canon VC-C4/VC-C4R Camera, a PC acting as server and another one acting as client, see figure 1.

The VC-C4 is a desktop camera, while VC-C4R is a reverse model suited for ceiling mount. Both cameras have the ability to connect to a computer via serial port. The VC-C4 and VC-C4R handles a set of commands received by the serial port, through which we can control pan, tilt, zoom and more, see figure 2. A list of all available commands are described in the appendix.

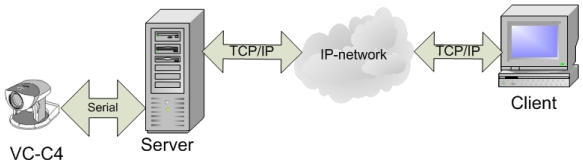

Figure 1: Connection schema.

1.1 Description of the problem

The aim of the project is to control the Camera in real time, in other words the application is time-critical, so we have to look into the factors that are time consuming. We can easily locate time-critical parts of the application into three different parts, serial communication, network latency and data processing. Serial communication is limited by hardware and the amount of data sent to an received from the camera is pre-programmed by the manufacturer, so there is not much we can do about it.

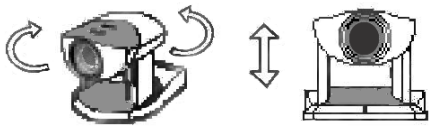

Figure 2: Tilt and Pan.

1.2 Overview of the client

The network-latency depends on distance and speed, outside the LAN (Local Area Network), IP packets travel through gateways and routers on the

Internet, which introduce transit delays, by reducing the amount of data, we can reduce overall latency. Now the last part which is data processing depends fully on the algorithms in our application, we have to use fast algorithms and process as little data as possible.

The client application can be implemented in any programming language, as long as it communicates using TCP/IP. A Friendly user-interface makes it easy for the user to navigate, and attaching a joystick, steering pad or any input device is no problem at all.

Following is the client list of functions:

Command	**Type**	**Description**
Connect	Menu	Connect to server
Disconnect	Menu	Disconnect from server
Exit	Menu	Release resources and Exit
Help	Menu	Show About dialog box
North east	Button	Tilt and pan north east
North	Button	Tilt north
North east	Button	Tilt and pan north west
East	Button	Pan east
Home	Button	Center
West	Button	Pan west
South east	Button	Tilt and pan to south east
South	Button	Tilt south
South west	Button	Tilt and pan to south west
Zoom	Slider	Zoom in and out
Focus	Slider	Set manual focus
Auto Focus	Check box	Set auto focus
Save	Check box	Save camera position
P1-P9	Buttons	Nine saved positions
Set Time	Button	Adjust time
Disp. Time	Button	Display time
Remote	Button	Activate remote control
Power	Button	Turn camera on and off
IP address	Edit box	Server IP address
Port Nr.	Edit box	Set network port

We also have three command groups placed in boxes, to control exposure and white balance:

AE auto		
Command	**Type**	**Description**
Black light adjust	Check box	Auto adjust light
AE reference	Slider	Adjust brightness in AE mode
Flickerless	Check box	Flicker less AE see appendix 1
AE manual		
Iris	Slide	Adjust iris
Gain	Slider	Adjust gain
Shutter	Slider	Adjust shutter
White balance		
Auto White Balance	Check box	Set Auto white balance
White Balance	Slider	Adjust white balance

1.3 Overview of the server

The Interface of our server is less sophisticated, in fact the user has only two options, either the server is running and the user has the option to stop it, or the server is not running and the user can start it by pressing the (Run Server) button.

For security reason the server can be set to accept only one particular IP-address, the computer with this IP then will be the only one allowed to send control commands to the server. We also bind it to a specific network port of our choice.

The Server can only connect one client at a time, if one client is connected the next connection will be rejected. On the other hand sending control signals to the server from two different clients simultaneously is not what we want to achieve, this will only lead to unpredictable results.

Following is the server list of functions:

Command	**Type**	**Description**
Run Server	Button	initialize network
Disconnect	Button	Disconnect and release resources
IP address	Edit box	Client IP address
Port Nr.	Edit box	Set network port

2 Planning phase

The project goal is to be able to control a camera in real time, thus the loss of any performance will result in the loss of the desired effect. Therefor we have to be careful in choosing the proper tools and using them in the right way.

First we start by looking at some programs addressing similar problem, one of these programs is VCC4VCSample. The drawback of this program is that it did not implement most of the camera features, the user-interface has only nine buttons to control directions and power on/off. The second drawback is that it has no network support, in other words you can not control the camera remotely.

Another program is VCC4CONT. This one is much better than the previous mentioned one, it has a nice user-interface and more features of the camera is used, but unfortunately still no network support.

Next step is to gather all the information about technical parts involved in the project, and choose the most suitable solutions for it.

In my search I found out the following :

1. We have to use communication with the camera via a serial port, that is the way this camera is intended to be used see figure 3. A serial port

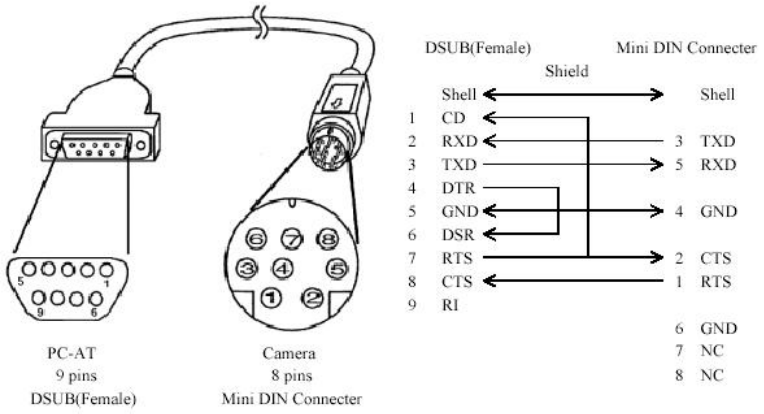

Figure 3: Connection schema.

   is an interface, with which information is transferred in or out one bit at a time, this is accomplished using the RS-232 standard over simple cables connecting the computer to a device.

The three most important abbreviations are, CTS (Clear To Send), TxD (Transmitted Data),RxD (Received Data) and GND (Ground).

2. In choosing the Internet Protocol I did not have the same limitation as in choosing an Interface to the camera, in this case the camera had no knowledge where the commands were coming from. I evaluated two different protocols namely TCP (Transmission Control Protocol) and UDP (User Datagram Protocol), at the end I chose TCP/IP for the following reasons:

   With only 4 header fields of which two are optional, UDP provides a faster transfer of data. UDP provides no error recovery, for this reason it is sometimes expanded to Unreliable Datagram Protocol. UDP applications must generally be willing to accept some loss, errors or duplication. UDP is connectionless, it means that a datagram can be sent at any moment without prior advertising, negotiation or preparation. Just send the datagram and hope the receiver is able to handle it.

   On the other hand being connection-oriented with TCP means that before actually transmitting data, you must open the connection between the two end points. The data can be transferred in full duplex (send and receive on a single connection). When the transfer is done, you have to close the connection to free up system resources. Both ends know when the session is opened (begin) and is closed (end). The data transfer cannot take place before both ends have negotiated the connection. The connection can be closed by either side; the other side is notified. Provision is made to either close gracefully or just to abort the connection.

2.1 Tools

In choosing the development tools, we have to take in to consideration the availability of documentation, because when we need help the more popular tool we choose, the easer it is to get help, from mailing lists, web sites, books and examples.

The second issue to consider in choosing our tool is the tool's ability to help us achieve our goal.

With these points in mind, I began my searching for the proper tools, by comparing different assemblers and found MASM32 (Microsoft assembler) to be the most common implementation, it has been used since the 1980's and has so many enthusiasts around the world.

When it came to the IDE(Integrated Development Environment) it did not make any difference, because the project is not dependent on the it, meaning that it could be replaced in mid-project without effecting project functionality. So I decided to use RADasm, because it has code highlighting for MASM32 and visual resource editor to build a user-interface.

2.1.1 Compiler MASM32

MASM32 is an assembler developed by Microsoft easy to use, with support for some high level structure, like if-, while-statements et cetera.

Since MASM32 is a 32 bit compiler it supports windows programming, hence a programmer can invoke or call a windows routine.

The compiler includes a 32 bit linker(link) and a resource compiler (rc), to help build the final application.

A typical make file could look like this:

```
 @echo off
if exist #1.obj del #1.obj
if exist #2.rc del #2.rc
\backslash masm32\backslash bin\backslash ml /c /coff #1.asm
if errorlevel 1 goto error1
\backslash masm32\backslash bin\backslash rc /c /coff #2.asm
if errorlevel 1 goto error2
\backslash masm32\backslash bin\backslash Link /SUBSYSTEM:WINDOWS #1.obj #2.obj
if errorlevel 1 goto error3
goto next
:error1
```

```
echo error in creating resource file
goto next
:error2
echo error in assembling source file
:error3
echo error in linking the object files
:next
pause
```

2.1.2 IDE RADasm

RADasm is an Editor with support for highlight of code and indentation, it also has a very nice visual resource editor to help build the user-interface, se figure 4.

RADasm has many features like building a project file for all the different code files involved in a project, it also hooks to a compiler to build and run the program from with in the IDE.

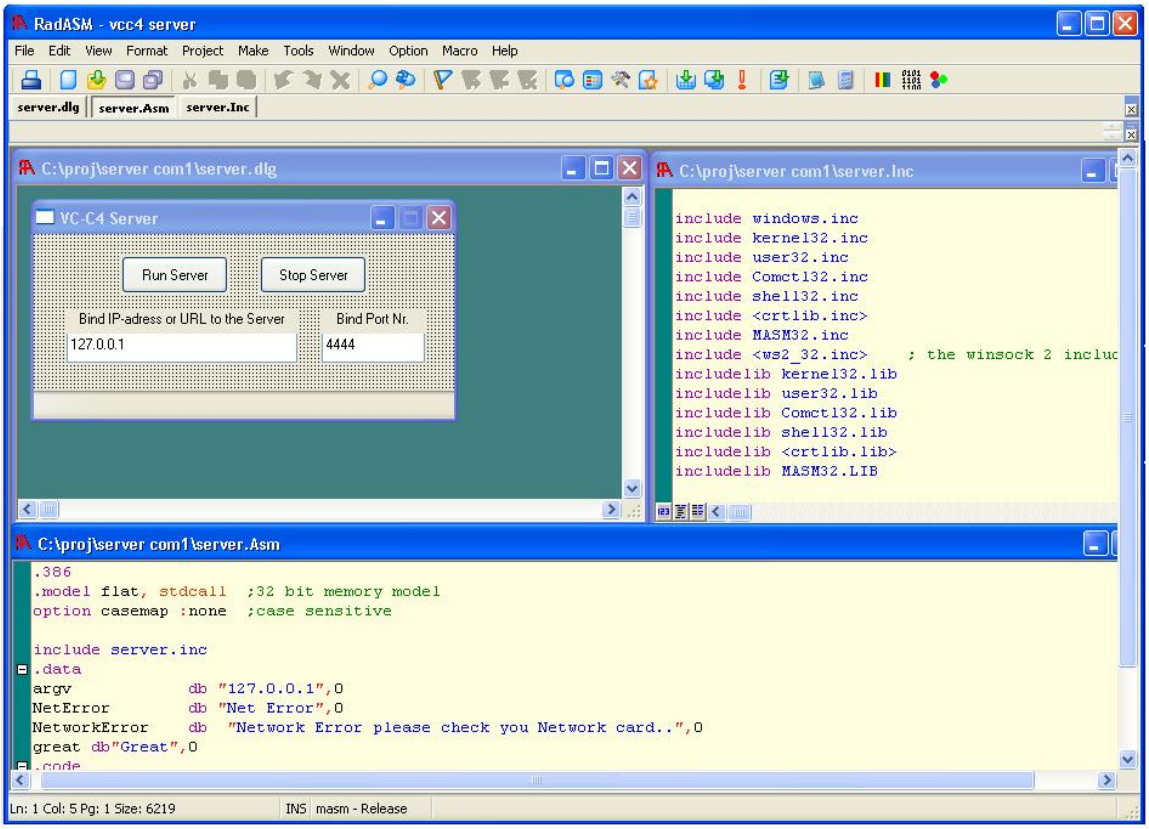

Figure 4: Connection schema.

3 Implementation phase

In our implementation we build every part as a unit, and then connect all units together.

First we built a small network consisting of two computers, the first computer would be our server, where the target camera is connected to. The second computer is a client, which will request the connection.

3.1 TCP/IP

TCP/IP protocol is connection-oriented between two sides, each has its own IP and port number. One side is called the server, the other side the client.

We use Winsock (Windows Sockets) in the Windows API (Application Programming Interface) that deals with networking. A socket is our handle to a connection, both client and server has a handle to a connection. A socket is a two-way connection, and they are not platform-specific, that is how a Windows and a Unix machine can communicate with each other.

In our project we use the API function bind to bind a socket in our server to a specific IP and port, a combination of bind IP and port by the server is the only one granted connection, se figure 5.

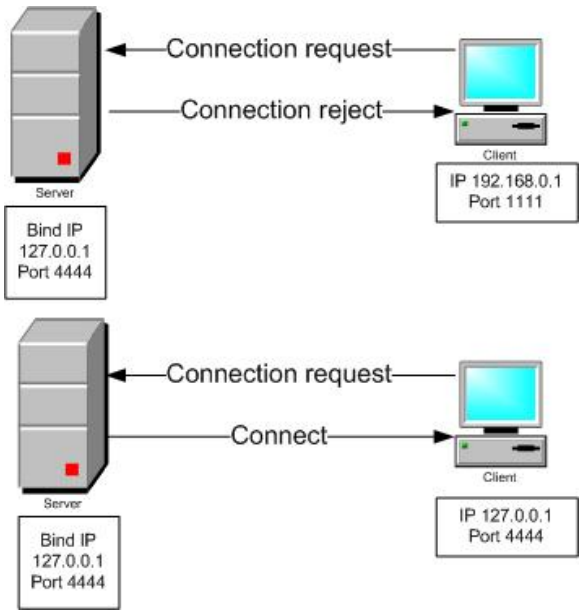

Figure 5: Bind IP and port.

The following code sample shows how to create a connection to the server:

```
;--------------------------------------------------------------------------
; <SetServerSockAddr>
;--------------------------------------------------------------------------
; Parameters
; pSockAddr pointer to the sockaddr_in structure to fill
; portNumber server port number
; Return value
;   Nothing
SetServerSockAddr proc pSockAddr:dword, portNumber:dword

mov edx, [pSockAddr]
mov ecx, [portNumber]
xchg  cl, ch ; convert to network byte order

mov [edx][sockaddr_in.sin_family], AF_INET
mov [edx][sockaddr_in.sin_port], cx
mov [edx][sockaddr_in.sin_addr.S_un.S_addr], INADDR_ANY

ret
SetServerSockAddr endp
```

3.2 Serial port

For serial communication we use API functions CreateFile, ReadFile and WriteFile, they open a port as if it is a file, and gives us a handle to the buffert where we can send data to or receive data from. Data we place in the buffer using WriteFile will be received by the camera, and by using ReadFile we receive response.

The following code sample shows how the server connects to the camera:

```
OpenPort proc comX:dword
invoke CreateFile,addr comX,GENERIC_READ or GENERIC_WRITE,0,0,OPEN_EXISTING,0,0
mov hcomm,eax
.IF eax==INVALID_HANDLE_VALUE
mov eax,1
mov herror, offset g_SPErorr
.ELSE
mov eax,0
.ENDIF
ret
OpenPort endp
```

By calling function CreateFile we recive a handel in hcomm, this handle can be used later communicate with the camera.

The following code sample shows how to read from and write to the camera:

```
WriteF proc
invoke WriteFile,hcomm,addr Header,sizeof Header-1,addr n,NULL
ret
WriteF endp

ReadF proc
invoke ReadFile,hcomm,addr data,sizeof data-1,addr n,NULL
ret
ReadF endp
```

The serial port converts each byte to a stream of ones and zeroes before sending and back again to byte when receiving data. An electronic chip called a UART (Universal Asynchronous Receiver/Transmitter) is actually the one that does the conversion.

3.3 Server

The server application has a very simple interface, two buttons one for putting the server online and the other one for taking it of line, as well as two edit boxes to specify the IP address of the remote client and the port to be used, se figure 6.

When a user presses the Run server button, we check for the camera via serial port, if camera exist we initialize it by adjusting the head to be centered, then we start a new socket, the socket is bind to the IP and port numbers specified in the edit boxes, finally the program gets into a standby position, waiting for a client to connect. As soon as connection is request by a client, we verify the client IP and port used to connect , if the IP and port matches the one entered into the edit boxes, access is granted by the server and connection is established.

The server receives single character from the client, each character corresponds to a command that is recognized by the camera, the server then sends the command to the camera. After the command has been issued, the camera sends a verification signal to the server.

Figure 6: Server.

Following code shows one sample out of 40 implemented commands:

```
.if al=="T"    ; letter corresponding to command
mov eax,0EF313260h ; command for pan right tilt down
mov [esi+4],eax    ; we put it in buffer
invoke WriteF    ; and send it to the serial buffer
...
```

3.4 Client

The client program has a more complicated user-interface, to give the user control over as many functions as possible. The client program has nine direction buttons, for directional head movements, two sliders for adjusting zoom and focus, buttons to save nine different positions, sliders to control light, brightness and some buttons for time and power, se figure 7.

When the User presses Connect from the menu, a connection request is sent to the server which IP address is in the edit box. If the server accepts the client IP, connection will be established and access to the device is granted by the server. Now the client program is ready to send commands to the camera.

Every command is saved as a single character in order to save band-width and to achieve a fast transfer rate. Pressing "Disconnect" or "Exit" will terminate the connection and give control back to the server, "Exit" will terminate the client application as well.

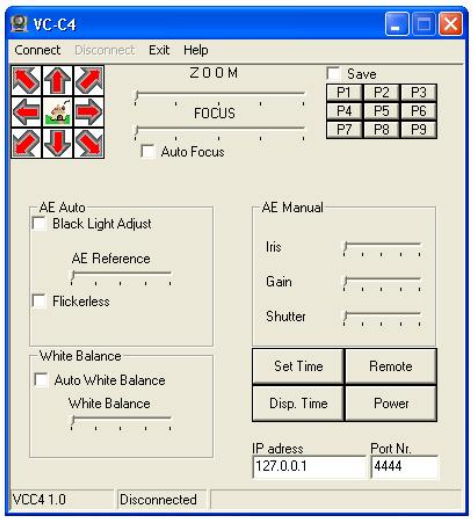

Figure 7: Client.

Following code sample shows how the decoding of commands is done:

```
c_Power db "A",0 ;Define Char A to turn camera on and off
c_Home db "H",0 ;Define Char H to position set position home
c_UL db "T",0 ;Define Char T to move UP and LEFT
...
.if eax == B1 ; Is button B1 pressed?
```

```
mov eax,offset c_UL ; Bring the definition of command
mov ebx, sizeof c_UL ; and the size
invoke WriteData,eax,ebx ; send it to server
...
```

4 Test phase

I divided the whole project into small units, test was made in the same fashion, each and every module was tested individually.

First of all I wrote a small application to communicate with the camera, using serial port we send commands to and receiving response from the camera. After analyzing the results I found out that receiving response take more time then sending, so I decided to receive the response in a different thread. This way I could cut the time spent by send and receive to less then half.

I moved next to the internet protocol, I wrote a small application that generated random data, then send it back and forth between two computers, and at the end the data is been analyzed by the application, analysis showed that looking for received data took more time then working with it, so I moved this part as well to a different thread.

Both threads will notify the main thread when data is available, this way our main thread will have more time to work on data received.

I merged the two applications, after correcting some bugs I run the test a few more times, to compeer sent and received data.

From this point I began to build the final application, of course I hade to optimize every module for speed and minimize the use of CPU and the size of data transfer.

Finally I run some more tests using my test applications, and more analysis of data is made to the final application, at this point the program was ready and needed only a friendly user-interface.

After building the user-interface, test had to be done manually to all the components in both server user-interface and Client user-interface, and the application was ready to be used.

5 Conclusion

In building such a time critical application, we need to consider using a low level programming language.

In this project there are three time critical moments:

1. The serial communication with the camera.
2. The network communication.
3. The data encoding and decoding done by the application.

The Remote Camera Controller system can be useful where we need to monitor several geographically separated areas from one place, by placing a camera in every location and remotely controlling them. It can also be used to stream pictures from hazardous areas, like areas contaminated with toxic gas or high degree of pollution.

References

Janet L. Axelson
2000
Serial Port Complete, lakeview research llc
ISBN: 0-9650819-2-3

Bob Quinn and Dave Shute
1995
Windows Sockets Network Programming, Addison-Wesley.
ISBN: 0-201-63372-8

Kip R. Irvine
2003
Assembly Language for Intel-Based Computers 4th ed., Prentice-Hall Inc.
ISBN: 0-13-091013-9

Charles Petzold
1999
Programming Windows 5th ed., Microsoft Press
ISBN: 1-57231-995-X

Forum For Assembler Language Programmers
`http://www.masmforum.com`
Last visited 20 May 2005

The Microsoft Developer Network
`http://msdn.microsoft.com`
Last visited 23 May 2005

Serial port (RS-232) Interfacing
`http://www.beyondlogic.org/index.html`
Last visited 10 May 2005

Winsock Programmer's FAQ
`http://tangentsoft.net/wskfaq`
Last visited 1 April 2005

Appendix: Control Command Tables

Following tables describes all the commands for communicating with Canon VC-C4 camera, via the serial port RS-232C interface:

Pedestal Control Command Table

Function	Meaning	Command	Parameter	Status
Pan Speed Assignment	To set running speed for Pan	0050h	008h 320h	None
Tilt Speed Assignment	To set running speed for Tilt	0051h	008h 26Eh	None
Pan Speed Request	To return present running speed for Pan	0052h	0h	008h 320h
Tilt Speed Request	To return present running speed for Tilt	0052h	1h	008h 26Eh
Pan Tilt Stop	To stop running of Pan/Tilt	0053h	0h	None
Pan Right Start	To start Pan running to right	0053h	1h	None
Pan Left Start	To start Pan running to left	0053h	2h	None
Tilt Up Start	To start Tilt running to up	0053h	3h	None
Tilt Down Start	To start Tilt running to down	0053h	4h	None
Home Position	To move Home position	0057h	None	Non
Pedestal Initialize 1	After Initialization, to move home position	0058h	0h	None
Pedestal Initialize 2	After Initialized, to move the origin position	0058h	1h	None
Pan Slowest Speed Request	To return the slowest speed of Pan	0059h	0h	008h
Pan Fastest Speed Request	To return the fastest speed of Pan	0059h	1h	320h
Tilt Slowest Speed request	To return the slowest speed for Tilt	0059h	2h	008h
Tilt Fastest Speed request	To return the fastest speed for Tilt	0059h	3h	26Eh
Pan Angle Pulse Ratio Request	To return coefficient of Pan angle conversion	005Bh	0h	2BF2h
Tilt Angle Pulse Ratio Request	To return coefficient of Tilt angle conversion	005Bh	1h	2BF2h
Pan Minimum Angle Request	To return minimum angle of Pan	005Ch	0h	7C87h (7A19h)
Pan Maximum Angle Request	To return maximum angle of Pan	005Ch	1h	8379h (85F7h)
Tilt Minimum Angle Request	To return minimum angle of Tilt	005Ch	2h	7EF5h (7CF0h)
Tilt Maximum Angle Request	To return maximum angle of Tilt	005Ch	3h	8320h (8059h)
Pan/Tilt Stop	To stop Pan/tilt running	0060h	00h	None
Pan/Tilt Start Stop	To start and stop Pan/tilt running	0060h	01h 22h	None
Pan/Tilt Angle Assignment	To move assign position of Pan/Tilt	0062h	XXXX, YYYYh	None
Pan/Tilt Angle Request	To return present position of Pan/Tilt	0063h	None	XXXX, YYYYh
Pan Movable Range Assignment	To assign movable limit for Pan	0064h	0h, PMIN, PMAX	None
Tilt Movable Range Assignment	To assign movable limit for Tilt	0064h	1h, TMIN, TMAX	None
Pan Movable Range Request	To return present position of Pan	0065h	0h	PMIN, PMAX
Tilt Movable Range Request	To return present position of Tilt	0065h	1h	TMIN, TMAX

Note: XXXX, YYYY means Pan angel, Tilt angle In Hexadecimal

PMIN, PMAX means Pan min. movable range, max. movable range in Hexadecimal

TMIN, TMAX means Tilt mini. movable range, max. movable range in Hexadecimal

The value inside ( ) means status value in the inverse mount type

Camera Control Command Table

Function	Meaning	Command	Parameter	Status
Camera OFF	To turn OFF power for Camera section	00A0h	0h	None
Camera ON	To turn ON power for Camera section	00A0h	1h	None
Focus Auto	To change mode of focus to AF	00A1h	0h	None
Focus Manual	To stop and change of focus to Manual	00A1h	1h	None
Focus Near	To move to near focus	00A1h	2h	None
Focus Far	To move to far focus	00A1h	3h	None
Focus Position Assignment	To move to focus position assigned	00B0h	XXXXh *1	None
Focus Position Request	To return present focus position	00B1h	0h	XXXXh *1
One Push AF	After adjustment of focus, change to AF	00B1h	1h	None
Focus Range Request	To return movable range of focus	00B1h	2h	XXXXXXXXh *2
Zoom Stop	To stop zoom operation	00A2h	0h	None
Zoom Wide	To zooming to wide	00A2h	1h	None
Zoom Tele	To zooming to tele	00A2h	2h	None
Zoom Hi Wide	To zooming to wide high speed	00A2h	3h	None
Zoom Hi Tele	To zooming to tele high speed	00A2h	4h	None
Zoom Position 1 Assignment	To move to zoom position assigned	00A3h	00h 80h	None
Zoom Position 1 Request	To return present zoom position	00A4h	None	00h 80h
Zoom Position 2 Assignment	To move to zoom position assigned	00B3h	0000h Max	None
Zoom Position 2 Request	To return present zoom position	00B4h	0h	0000h Min
Zoom Speed Assignment	To assign running speed of zoom	00B4h	”31h”, 0h˜7h	None
Zoom Speed Request	To return present running speed	00B4h	2h	0h 7h
Zoom Maximum Request	To return maximum movable position	00B4h	3h	XXXXh *3
Backlight Compensation OFF	To eliminate compensation of backlight	00A5h	0h	None
Backlight Compensation ON	To compensate backlight	00A5h	1h	None
Exposure Auto	To control exposure automatically	00A5h	2h	None
Exposure Manual	To control exposure manually	00A5h	3h	None
AE Lock OFF	To cancel AE lock ON	00A5h	40h	None
AE Lock ON	To lock the exposure of AE mode	00A5h	41h	None
Shutter Speed Program	To change shutter speed to program mode	00A8h	0h	None
Shutter Speed 1/60 (PAL: 1/50)	To change shutter speed to 1/60 (PAL: 1/50)	00A8h	1h	None
Shutter Speed1/100 (PAL: 1/120)	To change shutter speed to 1/12 (0PAL: 1/100)	00A8h	2h	None
Shutter Speed Assignment	To assign shutter speed	00A5h	”35h”, 00h˜19h	None

1. 4 figures Hexadecimal value within the request of focus limit

2. The focus range changes according to the zoom position. The value indicates Max/Min 4 figures in Hexadecimal

3. The zoom value is to set individually inherent value within 0000h FFFFh

System control Command Table

Function	Meaning	Command	Parameter	Status
Remote Control ON	To available remote controller	0080h	0h	None
Remote Control OFF	To inhibited remote controller	0080h	1h	None
Operation Status Request	To return information of operate status	0086h	None	3 byte
Extended Operation Status Request	To return extended information of operate status	0086h	0h	5 byte
Product Name Request	To return product name	0087h	None	VC-C4
ROM Version Request	To return ROM version of VC-C4	0088h	None	V5-16
Preset Set	To memory preset position	0089h	1h 9h	None
Preset Move	To move preset position	008Ah	1h 9h	None
Preset Status Request	To request preset status	008Bh	None	2 byte
Extended Preset Status Request	To request extended preset status	008Bh	0h	3 byte
Remote Controller Through Setting	To set ON/OFF of remote controller through	008Dh	0h 1h	None
LED Normal Display	To set normal display of LED	008Eh	0h	None
LED Forced Control	To set forced ON/OFF of LED	008Eh	1h 4h	None
Cascade OFF	To release cascade connection OFF	008Fh	0h	None
Cascade ON	To connect cascade connection ON	008Fh	1h	Non
Host Control Mode	To control by host computer	0090h	0h	None
Local Control Mode	To control by remote controller	0090h	1h	None
Screen Control	To set screen display of date, time, characters	0091h	00h 09h	None
Display Character Data Assignment	To assign character data of display	0091h	1h XX,Y,DD	None
Display Character Data Request	To request character data of display	0091h	2h XX,Y	2 byte
Display Date Setting	To set display date (yy/mm/dd)	0091h	3h YY,MM,DD	None
Display Date Request	To request display date (yy/mm/dd)	0091h	4h	YY,MM,DD
Display Time Setting	To set display time (hh/mm/ss)	0091h	5h,HH,MM,SS	None
Display Time Request	To request display time (hh/mm/ss)	0091h	6h	HH,MM,SS
Turning On Time Request	To request total accumulated turn on time	0092h	0h 1h	0000h FFFFh
Default Setting	To initialize in status of factory setting	0093h	None	None
Command Termination Notification Setting	To set termination notification on/off	0094h	0h 1h	None
Global Notification Setting	To set global notification on/off	0095h	0h 1h	None
Pedestal Model Request	To return pedestal model (normal/inverse)	009Ah	0h	0h 1h
Camera Model Request	To return camera model (NTSC/PAL)	009Ah	1h	0h 1h

1. XX, Y, DD indicate X address, Y address, ,Character Data of Display in Hexadecimal
2. YY, MM, DD indicate Year, Month, Day in Hexadecimal
3. HH, MM, SS indicate Hour, Minute, Second in Hexadecimal